On TLS/SSL: Research and Practice

Devdatta Akhawe
At CA/Browser Forum
About Me

• Engineer in Product Security team at Dropbox
  – Opinions expressed today are my own and do not necessarily represent the views of my employer
• Previously, grad student at UC Berkeley
  – Working on web security, SSL, usability etc
• Also, editor of specs at W3C
  – Sub-resource Integrity and Sub-origins
Today

• Research on SSL Warnings and Errors
  – Large scale study of source of SSL errors in the wild
  – Proposals and ideas on how to mitigate these issues

• Experience deploying advanced SSL features at Dropbox
Part 1

Large Scale study of SSL errors
Let’s talk about TLS warnings
This is probably not the site you are looking for!

You attempted to reach reddit.com, but instead you actually reached a server identifying itself as a248.e.akamai.net. This may be caused by a misconfiguration on the server or by something more serious. An attacker on your network could be trying to get you to visit a fake (and potentially harmful) version of reddit.com.

You should not proceed, especially if you have never seen this warning before for this site.

Proceed anyway  Back to safety

Help me understand
The “bypass this certificate error” button ... is a UI disaster.

Those buttons are clicked 60% of the time by Chrome users.

Adam Langley
Google, Inc.
One Explanation:
Too many false warnings due to misconfigurations!
Hypothesis: Tragedy of the Commons with TLS Warnings
shared resource? user attention

Consumers?
browsers, servers, proxies
The Lump of Attention Model

Shared resource: Attention

Böhme and Grossklags *The Security Cost of Cheap User Interaction*
NSPW 2011
This is probably not the site you are looking for!

You attempted to reach reddit.com, but instead you actually reached a server identifying itself as a248.e.akamai.net. This may be caused by a misconfiguration on the server or by something more serious. An attacker on your network could be trying to get you to visit a fake (and potentially harmful) version of reddit.com.

You should not proceed, especially if you have never seen this warning before for this site.

Proceed anyway  Back to safety

Help me understand
How to proceed with this website loading error?
Nothing new
While warnings can be improved, a better approach may be to minimize the use of SSL warnings altogether.

Sunshine et al.  
*Crying Wolf* ....  
Usenix Security 2009
Where do we warnings come from?
Cert is fine; client is configured wrong
See Alice in Warningland and follow ups

Hello
SNI:reddit.com

Cert is invalid
(this talk)
Research not in this talk

• Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness
  – Me and Adrienne Porter Felt (Google)
  – Study from inside browsers on what are the warning click through rates and what sort of SSL errors are really common
Research not in this talk

See Adrienne’s talk at AppSecCali to see the follow on work on all sources of errors in client side and work on improving warning adherence.

That supersedes work in this paper.
Today

A large scale measurement of TLS certificate errors to look for opportunities to conserve user attention
Here's My Cert, So Trust Me, Maybe?
Understanding TLS Errors on the Web

Devdatta Akhawe / Bernhard Amann / Matthias Vallentin / Robin Sommer

UC Berkeley and International Computer Science Institute

(Published in 2013)
Outline

Data Collection

Methodology

Results
https://reddit.com

- Issued by ACME, which is issued by ...
- Expiration
- Name/Length
- Constraints
- Issued for: CN & AltName

SNI:reddit.com
10 networks running the Bro network monitor
300K users
9 Months of data
3.9B connections to port 443
Outline

Data Collection

Methodology

Results
Not Implemented in OpenSSL

Need to use Browser libraries!
Chain Validation

For each certificate in chain:

- Is cert expired?
- Is cert revoked?
- Name/length constraints?
Name Validation

- Is the Cert really for intended website?
- Attacker can always get cert for attacker.com
- Reuse the browser code
- Compare saved SNI with the certificate
- OpenSSL introduced a name match function in 2013!
Outline

Data Collection
Methodology
Results
1.54% connections with errors, presumably false warnings
If an actual attack occurs once in a million connections, 15400 false warnings for 1 real warning

99.9994%

false warnings

Assuming no client-side config errors, which would make this number even worse
## Error Breakdown

<table>
<thead>
<tr>
<th>Error</th>
<th>Connections</th>
<th>Unique Certificates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unknown Issuer</td>
<td>70.51%</td>
<td>5,027</td>
</tr>
<tr>
<td>Self Signed Certificates</td>
<td>2.99%</td>
<td>6,126</td>
</tr>
<tr>
<td>Expired Certificates</td>
<td>7.65%</td>
<td>21,522</td>
</tr>
<tr>
<td>Name Validation Errors</td>
<td>18.82%</td>
<td>12,146</td>
</tr>
</tbody>
</table>

Untrusted issuers remain a big problem.
1. Free and easy certificates offered by CAs such as StartSSL, Lets Encrypt are valuable.

2. Mechanisms such as DANE & Convergence have value due to tremendous usability benefits.
Domains Secured by Let’s Encrypt (15 Feb 2016)

- 2,493,591 domains
- 149,270 domains

Only Issued by Let’s Encrypt | Also Issued By Other CA(s)
A majority of certificates used in erroneous certificates correspond to expired and name validation errors.

<table>
<thead>
<tr>
<th>Error</th>
<th>Connections</th>
<th>Unique Certificates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unknown Issuer</td>
<td>70.51%</td>
<td>5,027</td>
</tr>
<tr>
<td>Expired Certificates</td>
<td>7.65%</td>
<td>21,522</td>
</tr>
<tr>
<td>Name Validation Errors</td>
<td>18.82%</td>
<td>12,146</td>
</tr>
</tbody>
</table>
Expired Certificates

- Expired Certs common in the long tail
- 50% of expired certs used only 4 times
- 75% of expired certs used only 12 times

- 25% of all expired certificates accessed only for a week after expiry
- Presumably, renewed after that
Use a non-blocking infobar to warn for certificates expired in the last week.
Name Validation Errors
# Name Validation Errors

<table>
<thead>
<tr>
<th>Error</th>
<th>Connections</th>
<th>Unique Certificates</th>
</tr>
</thead>
<tbody>
<tr>
<td>WWW Mismatches</td>
<td>1.17%</td>
<td>7.92%</td>
</tr>
<tr>
<td>Multiple Names</td>
<td>1.21%</td>
<td>0.03%</td>
</tr>
<tr>
<td>Relaxed Match</td>
<td>50.40%</td>
<td>7.24%</td>
</tr>
<tr>
<td>Relaxed Match with WWW</td>
<td>51.54%</td>
<td>13.87%</td>
</tr>
<tr>
<td>TLD Match</td>
<td>56.93%</td>
<td>29.73%</td>
</tr>
</tbody>
</table>
# Name Validation Errors

<table>
<thead>
<tr>
<th>Error</th>
<th>Connections</th>
<th>Unique Certificates</th>
</tr>
</thead>
<tbody>
<tr>
<td>WWW Mismatches</td>
<td>1.17%</td>
<td>7.92%</td>
</tr>
<tr>
<td>Multiple Names</td>
<td>1.21%</td>
<td>0.03%</td>
</tr>
<tr>
<td>Relaxed Match</td>
<td>50.40%</td>
<td>7.24%</td>
</tr>
<tr>
<td>Relaxed Match with WWW</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TLD Match</td>
<td>30.95%</td>
<td>20.73%</td>
</tr>
</tbody>
</table>

User wants to connect to paypal.com and cert says www.paypal.com
Tolerate WWW mismatches or show a different “low-risk” warning.
User wants to connect to foo.bar.test.com and cert is for *.test.com

<table>
<thead>
<tr>
<th>Error</th>
<th>Connections</th>
<th>Unique Certificates</th>
</tr>
</thead>
<tbody>
<tr>
<td>WWW Mismatches</td>
<td>1.17%</td>
<td>7.92%</td>
</tr>
<tr>
<td>Relaxed Match</td>
<td>50.40%</td>
<td>7.24%</td>
</tr>
<tr>
<td>Relaxed Match with WWW</td>
<td>51.54%</td>
<td>13.87%</td>
</tr>
<tr>
<td>TLD Match</td>
<td>56.93%</td>
<td>29.73%</td>
</tr>
</tbody>
</table>
Move to a relaxed matching algorithm that accepts multiple levels for an asterisk.
User wants to connect to foo.bar.com and cert is for www.bar.com

<table>
<thead>
<tr>
<th>Error</th>
<th>Connections</th>
<th>Unique Certificates</th>
</tr>
</thead>
<tbody>
<tr>
<td>WWW Mismatches</td>
<td>1.17%</td>
<td>7.92%</td>
</tr>
<tr>
<td>Multiple Names</td>
<td>1.21%</td>
<td>0.03%</td>
</tr>
<tr>
<td>Relaxed Match</td>
<td>50.40%</td>
<td>7.24%</td>
</tr>
<tr>
<td>TLD Match</td>
<td>56.93%</td>
<td>29.73%</td>
</tr>
</tbody>
</table>
Use a low-risk warning for sub-domain mismatch to help focus user attention on the high-risk scenarios.
• We started off with 15400 false warnings.
• We first need to fix chain errors > 70%
  – Convergence or TOFU can help
• IF we do, then using our other tricks, the number of false warnings, in our data, drops off to 213 per million
  – Still 99.5% false warnings! 😞
  – But browser vendors can make these stronger
A clear opportunity exists to reduce unnecessary consumption of user attention budget and help focus attention on high risk scenarios
Part 2

Deploying TLS at Scale
Case Studies

HSTS includeSubDomains
OCSP Stapling
HSTS on UserContent
The Problem:

How to deploy HSTS includeSubDomains on dropbox.com?
• https://carousel.dropbox.com
• https://photos.dropbox.com
• https://www.dropbox.com
• https://block.dropbox.com
• ... all public sites support SSL ...
• http://cafemenu.corp.dropbox.com
• http://busschedules.corp.dropbox.com
• https://carousel.dropbox.com
• https://photos.dropbox.com
• https://www.dropbox.com
• https://block.dropbox.com
• ... all public sites support SSL ...
• http://cafemenu.corp.dropbox.com
• http://busschedules.corp.dropbox.com
dropbox.com
  HSTS: 3 years,
  includeSubDomains

corp.dropbox.com
  HSTS: 3 years,
  includeSubDomains

foo.corp.dropbox.com
  HSTS policy?
Not an exception

A lot of sites don’t set includeSubDomains on root URI

Allowing HSTS overrides with enterprise policy or some config would help massively
Case Studies

HSTS includeSubDomains

OCSP Stapling

HSTS on UserContent
OCSP stapling

From Wikipedia, the free encyclopedia
(Redirected from OCSP Stapling)

OCSP stapling, formally known as the TLS Certificate Status Request extension, is an alternative approach to the Online Certificate Status Protocol (OCSP) for checking the revocation status of X.509 digital certificates.\[1\] It allows the presenter of a certificate to bear the resource cost involved in providing OCSP responses by appending ("stapling") a time-stamped OCSP response signed by the CA to the initial TLS Handshake, eliminating the need for clients to contact the CA.\[2][3\]
A Good OCSP implementation

- Robust against CA responder failures
- Should not DoS the responder by mistake
- Check for invalid responses and alert
- Support arbitrary certificates and arbitrary responder URIs
- Robust against network failures and other failures
Implementing OCSP Stapling

• The core idea is simple: write a script that fetches the response and tells nginx about it.
• And then you worry about all the problems in the previous slide.
Implementing OCSP Stapling

• The core idea is simple: write a script that fetches the response and tells nginx about it
Using OpenSSL

- ocsp command returns non-zero even with success sometimes
- ocsp command is sensitive to argument ordering
- ocsp verification command returns with 0 whether or not the response is valid
  - We have to manually scan for “OK” in output!
This is not tenable for large scale deployments

Agreeing to must-staple with this foundation is too risky

Better OCSP stapling services, examples, packages would help. Default nginx support not ok.

Need “Report only mode” in browsers
Case Studies

HSTS include SubDomains
OCSP Stapling
HSTS on UserContent
The Problem:

Sites host untrusted user content on a separate domain

Can we turn on HSTS?
Common: googleusercontent.com, *.github.io, and so on

Sites only link to it as https:

But users could directly link to it

Turning on HSTS will just break the user’s page if any fetch is over http
Thanks for listening!

evil@berkeley.edu
devd.me